IN THE SPECIFICATION

On page 2, after the first paragraph, please insert the following paragraph:

RELATED APPLICATIONS 2002 0083381

A2 This application is related to U.S. Patent Application No. 09/748, 825, filed December 26, 2000 and U.S. Patent Application No. 09/749,133, filed December 26, 2000.

2002 0083385

On page 11 please amend the second paragraph as indicated below:

A3 For one embodiment, memory 34 is DRAM (dynamic random access memory). For other embodiments, other types of semiconductor memory can be used. For yet other embodiments, memory 34 can comprise hard disk memory or nonvolatile memory. For one embodiment, memory 34 is external to host processor 22. For other embodiments, memory 34 can be included as part of host processor 22, forming a system on a chip, for example.

On page 14, please amend the second paragraph as indicated below:

A4 The write state machine 28 generates signals that initiate a strobe to bits of array 20 requiring ~~program or~~ programming or to a block of array 20 to erase. The write state machine 28 also generates signals to supervise strobe pulse width and associated timings. The write state machine 28 generates signals that control the data comparator 81. The write state machine 28 generates signals that request feedback from the data comparator 81 to determine pulse repetition control and provide an update to the status register 83. The write state machine 28 generates signals that initiate an address counter 63 for erase preconditioning or erase verify.

On page 15, please amend the second and third paragraphs as indicated below:

A5

The host processor 22 writes data to the array 20 generating write cycles over lines 26 (e.g., 102, 104, and 106) to transfer program commands and data to the command user interface circuitry 40. The command user interface circuitry 40 verifies the program commands, and queues the program commands, and address and data parameters, to the write state machine 28. The write state machine 28 performs the program operation by programming the specified data into the array 20 at the specified address.

Write state machine 28 includes circuitry 32 for enabling or disabling the special programming mode of flash memory 24. When the special programming mode is enabled by circuitry 32, the write state machine 28 prevents the internal verification of data written to memory array 20. Verification of program data is ~~the~~ a normal operation of flash memory 24.

On page 20, please amend the second paragraph as indicated:

A6

As shown in FIG. 7, the data that is sent equals data X, which indicates that a series of data words are being sent to flash memory 24. For one embodiment, the address equals the previous address, which remains the start address of the data stream. For another alternative embodiment, however, various addresses can be sent. The command user interface 40, write state machine 28, and special programming mode circuitry 32 remain in the data stream state 156 until a data stream termination condition is encountered.

On page 29, please amend the first paragraph as indicated:

Sub B5
A7

The write state machine 28 and special programming mode circuitry 32 cause the read operation with respect to flash memory array 20 to occur at the program verification voltage

levels. Thus, at process block 256, the write state machine 28 and special programming mode circuitry 32 cause a margined-sensing scheme to be used. An elevated read voltage is applied to the programmed cells. The current I_{PMRGN}, derived from a programmed margin bias read of the column containing the programmed cell, is fed into one of the sense amplifiers 117a-117p with reference current I_{PREF}, which is the current from the factory set program reference circuit 111. If I_{PMRGN} is less than I_{PREF}, then the particular sense amplifier of sense amplifiers [[of]] 117a-117p outputs a zero. If I_{PMRGN} is greater than I_{PREF}, then a sense amplifier of sense amplifiers 117a-117p outputs a logical one. These operations occur for all the bits of the word in parallel, using each of the sense amplifiers 117a-117p.

A7

On page 33, please amend the last paragraph as indicated below:

In order to exit the special programming mode, host processor 22 sends [[ ]] an exit special programming mode command to flash memory 24 via lines 26. The exit special programming mode command is received by the command user interface 40 of flash memory 24. The command user interface then enters state 160. In state 160, the command user interface is in the status state. At state 160, the output multiplexer 45 outputs a status signal. At state 160, the flash memory array 24 is no longer in the special programming mode. Once the flash memory array is no longer in the special programming mode, the flash memory array 24 can perform its usual internal program verification operations.

A8

On page 34, please amend the last paragraph as indicated below:

In order to permanently disable the special programming mode, referring to Figure 7, the [[ ]] process flow moves to state 162. At state 162, the host processor 22 sends a disable special programming mode command to the command user interface 40 over

A9

lines 26. Process flow then moves to process block 164. At process block 164, the flash memory 24 permanently disables the special programming mode. In particular, at process block 164, the write state machine 28 and special programming mode circuitry 32 set an internal register or CAM (content addressable memory) that prevent the write state machine 28 and special programming mode circuitry 32 from ever further entering the special programming mode. When the special programming mode is permanently disabled, the end-user then can no longer enter the special programming mode. That means that even if a host computer 22 were to send a special programming mode command to the flash memory 24, the write state machine and circuitry 32 would not allow the flash memory array to enter the special programming mode. With the special programming mode disabled, ~~then~~ programming of the flash memory array would occur in its normal mode, which would include internal program verification for each programming of a data word.

A9

On page 36, please amend the second paragraph as indicated below:

FIG. 9A illustrates an alternative programming sequence of one embodiment. First the status register is read at process block 300. For one embodiment, when the status register reads 0, then the program (PGM) phase is ready to begin as shown in process block 302. Next, the data is written to the memory in process block 304. The data is sent continuously until a program time (X_{ns}) has expired (i.e. timed out) as shown in process block 306. Next, the last data <u>word</u> is checked in process block 308. If the last data word was not sent to the memory, then the program process repeats as shown in process blocks 304, 306. If the last data word was sent to the memory, then the program phase is ended by writing FFFF in the block address as shown in process block 310.

A10

On page 37, please amend the second paragraph as indicated:

A 11

FIG. 10 shows programming and verification procedures 320 for [[an]] a special programming mode that includes hashing. For the operation 320 shown in FIG. 10, the process flow shown in FIG. 7 generally applies, except for the addition of hashing by flash memory 24.

On page 40, please amend the first paragraph as indicated:

Sub B7

A 12

The process flow then moves to process block 354. At process block 354, the host processor 22 compares the hash value stored in flash memory array 20 with a hash value that the host processor 22 had stored in memory 34 to see if they are the same. The hash value stored in memory 34 is a result of a dynamic hashing of the data stream words stored in memory 34 that were sent by host processor 22 for programming into flash memory 24. In other words, host processor 22 executes the same hashing algorithm as flash memory 24 with respect to the data words to be programmed into flash memory 24. If the hash values stored in array 20 and memory 34 are the same, then process flow moves to process block 356, which means that the data words in the data stream have all been successfully programmed into flash memory 24. That is because a hash operation means that there is a high likelihood that the result of the hash operation is a unique number. There would only be an extremely small possibility that the hash values would be the same compare even though the data stream words had not been successfully programmed into flash memory 24. If the hash values ~~compare~~ are substantially the same at process block 354, there is a high probability that the data stream was successfully programmed into flash memory 24.

On page 41, please amend the last paragraph as indicated:

For an alternative embodiment of the invention, process flow 320 could include the hashing of status information stored within flash memory 24 in addition to the hashing of the programmed data words dynamically. The status information could include the status value stored in status register 83 as well as other status information. Host processor 22 would store in its memory 34 an expected hash value that would result from the hashing of the data stream words and expected status information from flash memory 24. The hashing of status information in addition to data words would allow the host processor 22 to check for correct operation by flash memory 24 in addition to the correct programming of the data stream. For example, if a blocking error occurred, the alternative hashing procedure might capture that error. If status information and data words are hashed, the write state machine 28 and the special programming mode <u>circuitry</u> 32 would oversee the running of the hash algorithm.

A13

On page 44, please amend the third paragraph as indicated:

For the process flow 402 of Figure 11, the host processor sends the required sequence of special programming mode commands to get the flash memory into state 414, which is the special programming mode state. Once the command user interface 40 and the flash memory 24 are in state 414, they remain in the special programming mode state until the host processor 22 sends [[a]] <u>an</u> exit special programming mode command to cause the flash memory to go to state 160 <u>of Figure 7</u>, which is the status state of command user interface 40.

A14